COMP 308 ARTIFICIAL INTELLIGENCE PART 8.4 – K-NEAREST NEIGHBOUR

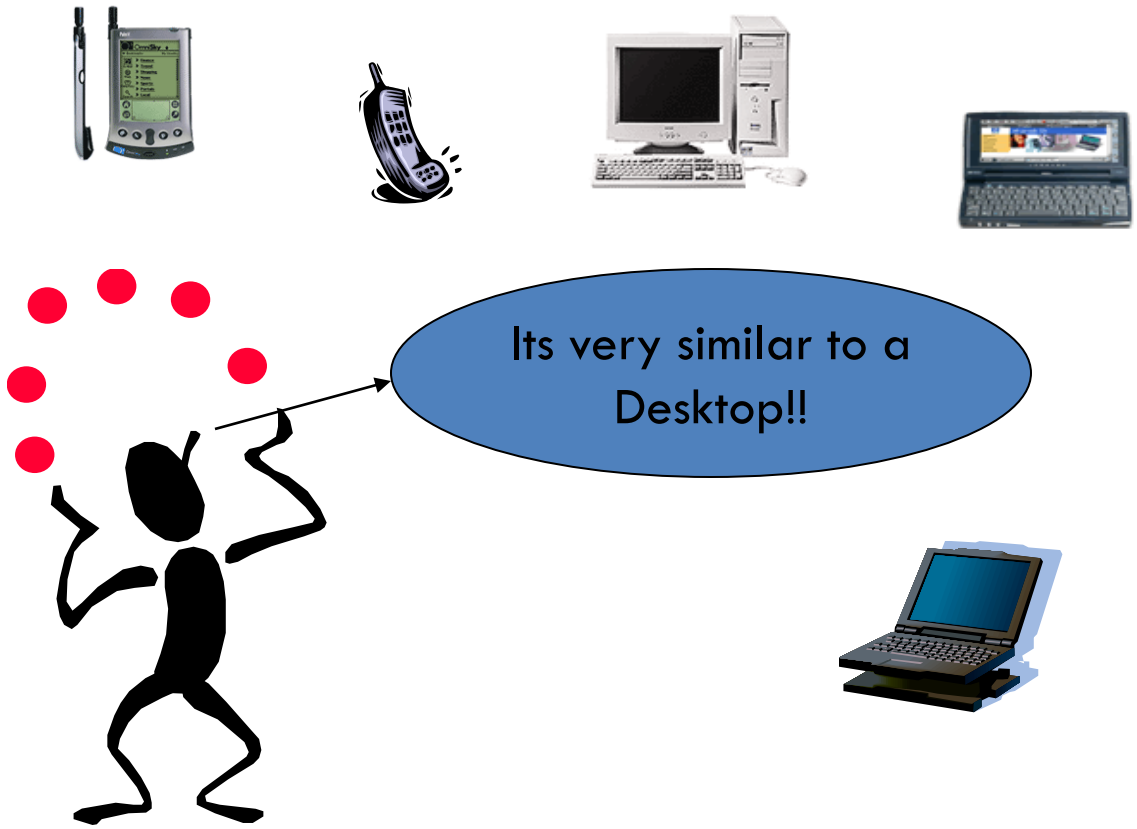
Njeri Ireri

Jan – April 2020

Overview

- Nearest Neighbor
- Locally weighted regression
- Case based reasoning (CBR)

Instance-based Learning



Nearest Neighbors

- **Nearest-neighbor** classifiers are based on learning by analogy, that is, by comparing a given test tuple with training tuples that are similar to it
- A **k-nearest-neighbor** often abbreviated K-NN, is an approach to data classification that estimates how likely a data point is to be a member of one group or the other depending on what group the data points *nearest* to it are in.
- Formally, the nearest-neighbor (NN) search problem is defined as follows:
  - given a set S of points in a space M and a query point $q \in M$, find the closest point in S to q

When to Consider Nearest Neighbors

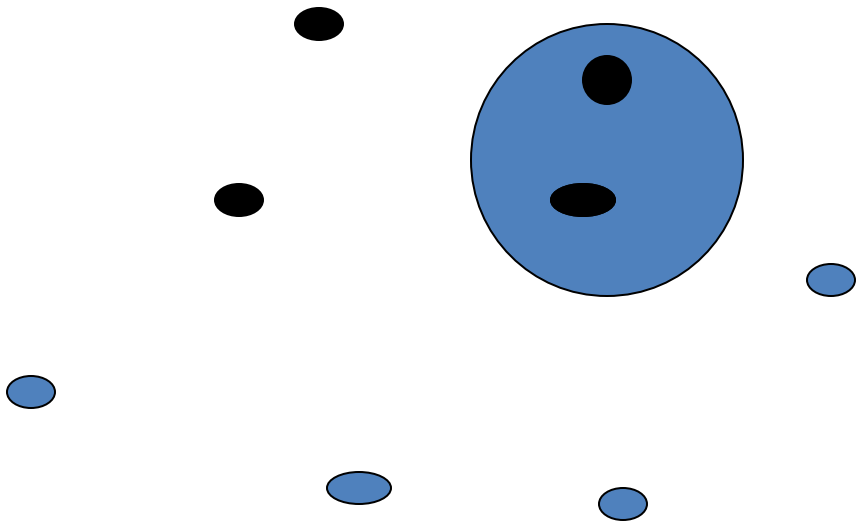
- Instances map to points in $\Re^n$ (where n-dimensional space)
- Less than 20 attributes per instance
- Lots of training data

Advantages :

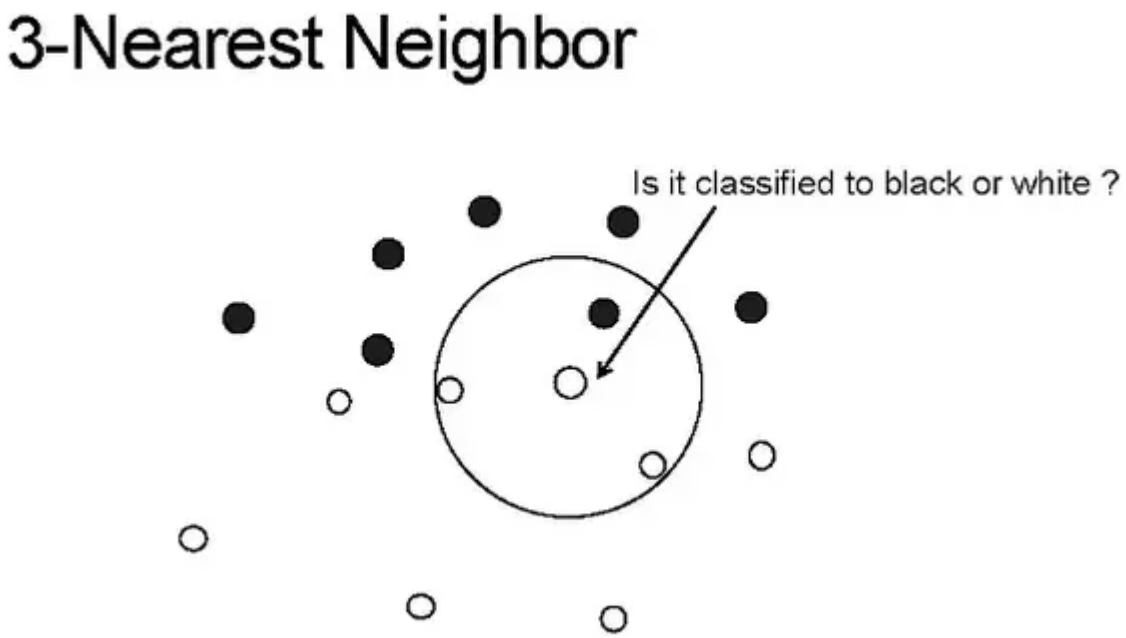
These are the strengths of the k-Nearest Neighbour machine learning technique:

- has the ability to model very complex target functions by a collection of less complex approximations
- It is easy to program this algorithm
- No optimization or training is required for this algorithm

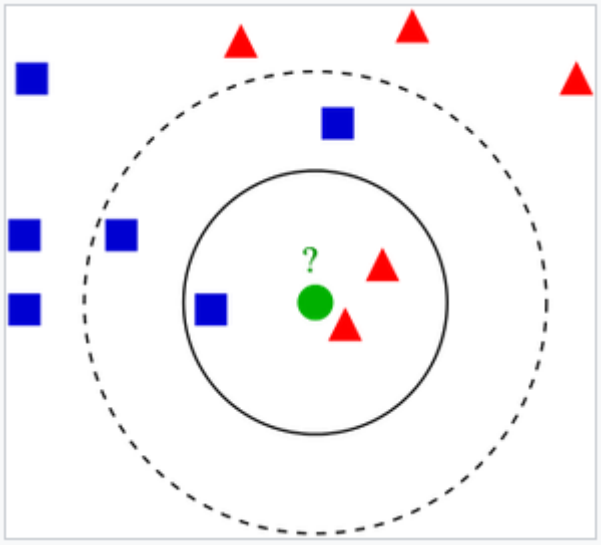
1-Nearest Neighbor



3 - Nearest Neighbors



Nearest Neighbors



Nearest Neighbors

- **Nearest neighbor search** (**NNS**), as a form of proximity search, is the optimization problem of finding the point in a given set that is closest (or most similar) to a given point. Closeness is typically expressed in terms of a dissimilarity function: the less similar the objects, the larger the function values. Formally, the nearest-neighbor (NN) search problem is defined as follows: given a set S of points in a space M and a query point $q \in M$, find the closest point in S to q.
- **Nearest Neighbor Analysis** is a method for classifying cases based on their similarity to other cases. In machine learning, it was developed as a way to recognize patterns of data without requiring an exact match to any stored patterns, or cases. Nearest neighbor analysis examines the distances between each point and the closest point to it, and then compares these to expected values for a random sample of points from a CSR (complete spatial randomness) pattern. CSR is generated by means of two assumptions: 1) that all places are equally likely to be the recipient of a case (event) and 2) all cases are located independently of one another.

When to Consider Nearest Neighbors

- It is robust to noisy training data. Provided with sufficiently large training dataset, it has been shown to be quite effective.
- Information can be incrementally added at run-time
- Information is never lost because the training examples are stored explicitly

Disadvantages:

- Slow at query time
- Easily fooled by irrelevant attributes

Instance Based Learning

Key idea: just store all training examples $<x_i, f(x_i)>$

Nearest neighbor:

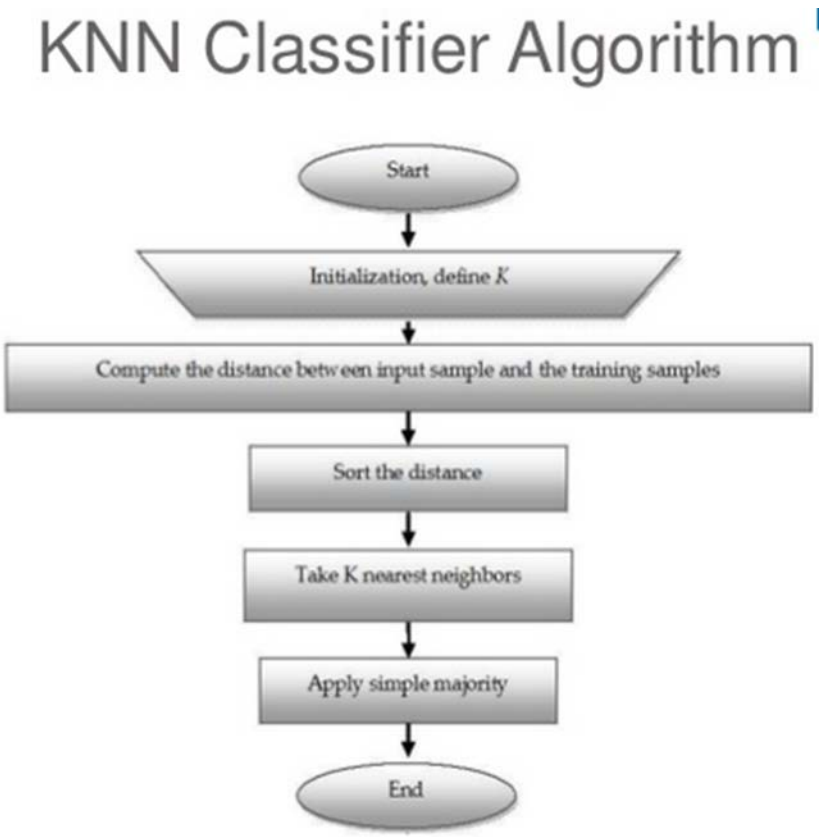
- Given query instance x_q, first locate nearest training example x_n, then estimate $\hat{f}(x_q)=f(x_n)$

K-nearest neighbor:

- Given x_q, take vote among its k nearest neighbors (if discrete-valued target function)
- Take mean of f values of k nearest neighbors (if real-valued) $\hat{f}(x_q)=\sum_{i=1}^{k} f(x_i)/k$

Note: $\hat{f}$ *- value returned by algorithm*

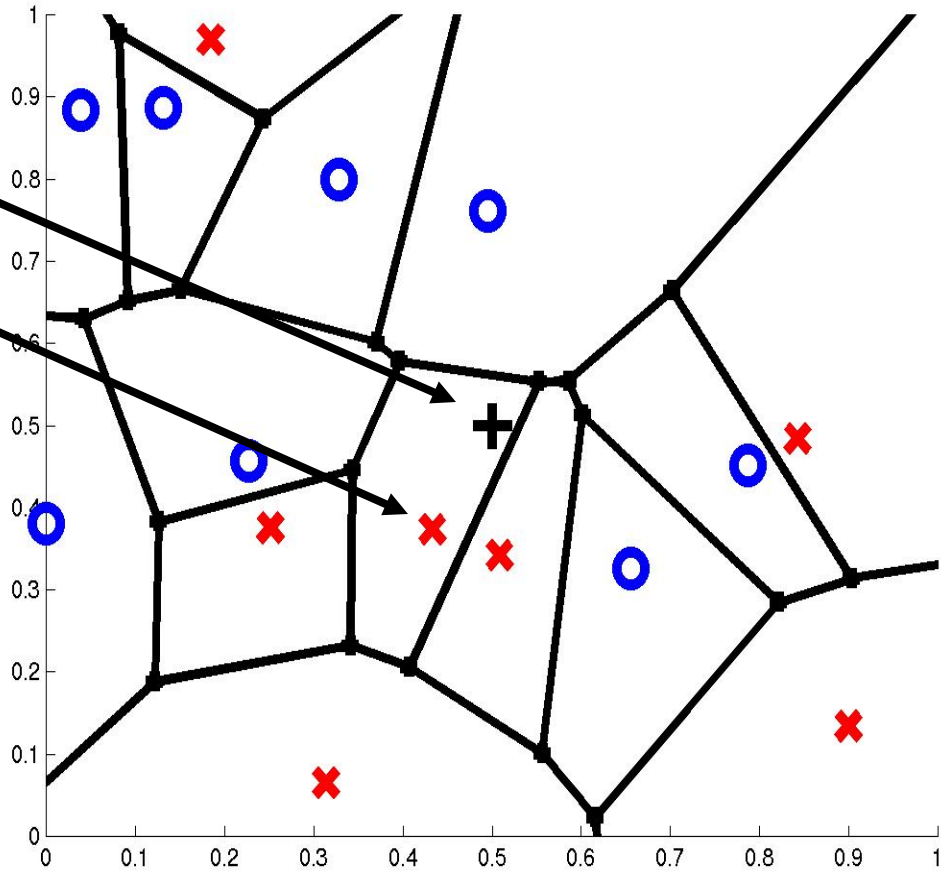
KNN classifier Algorithm



Voronoi Diagram

query point q_f

nearest neighbor q_i



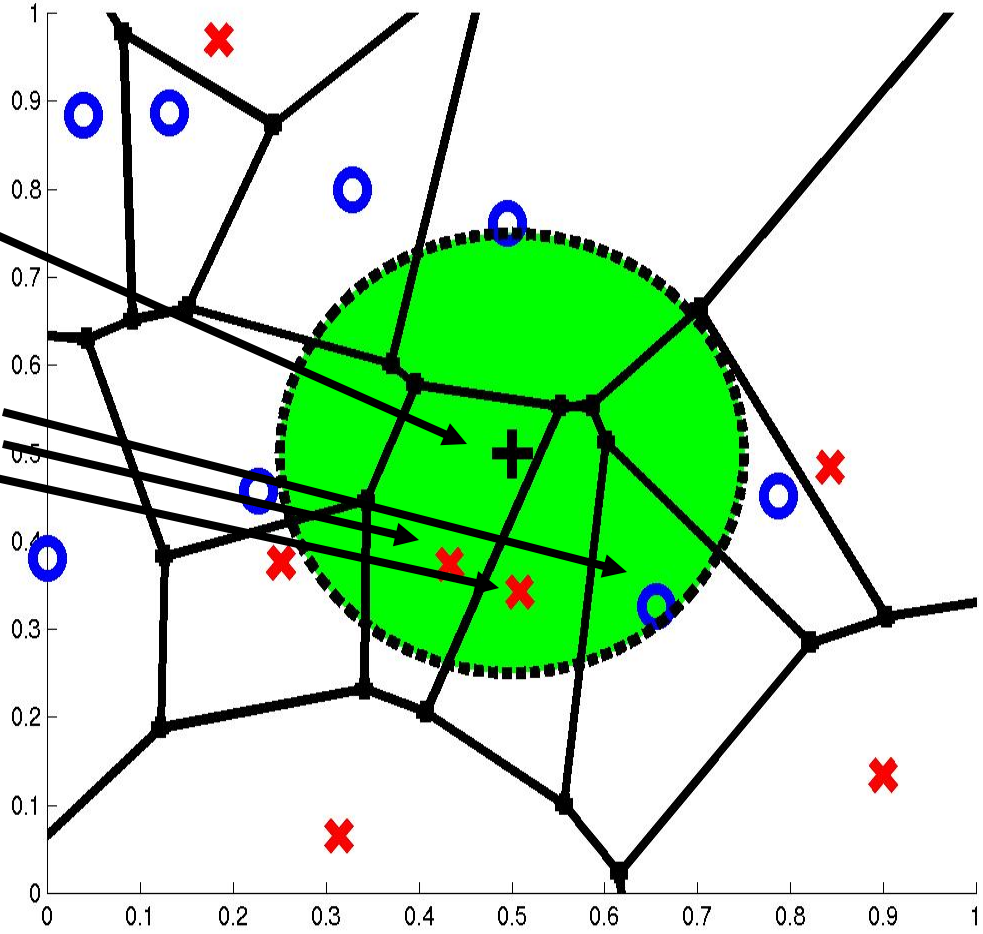
Example: complex 2-class problem(**x**,**o**)
using 2 attributes on x and y axis

3-Nearest Neighbors

query point q_f

3 nearest neighbors

2x,1o

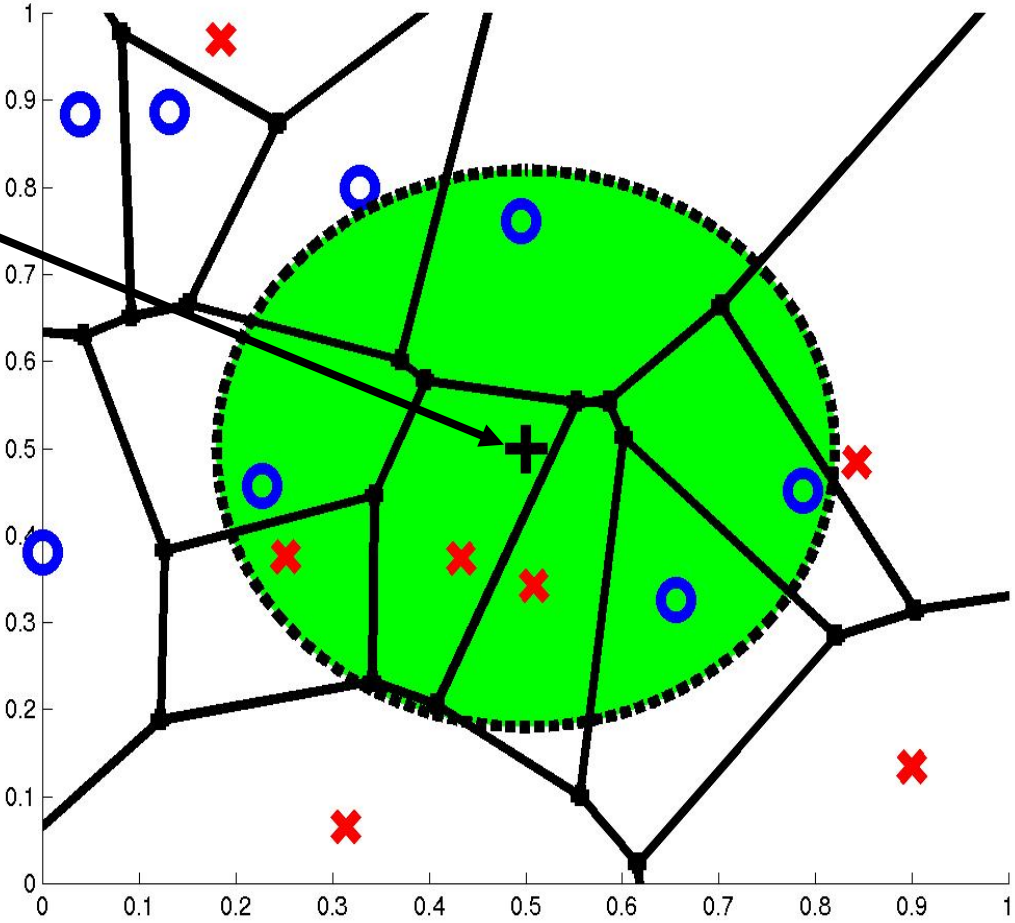


7-Nearest Neighbors

query point q_f

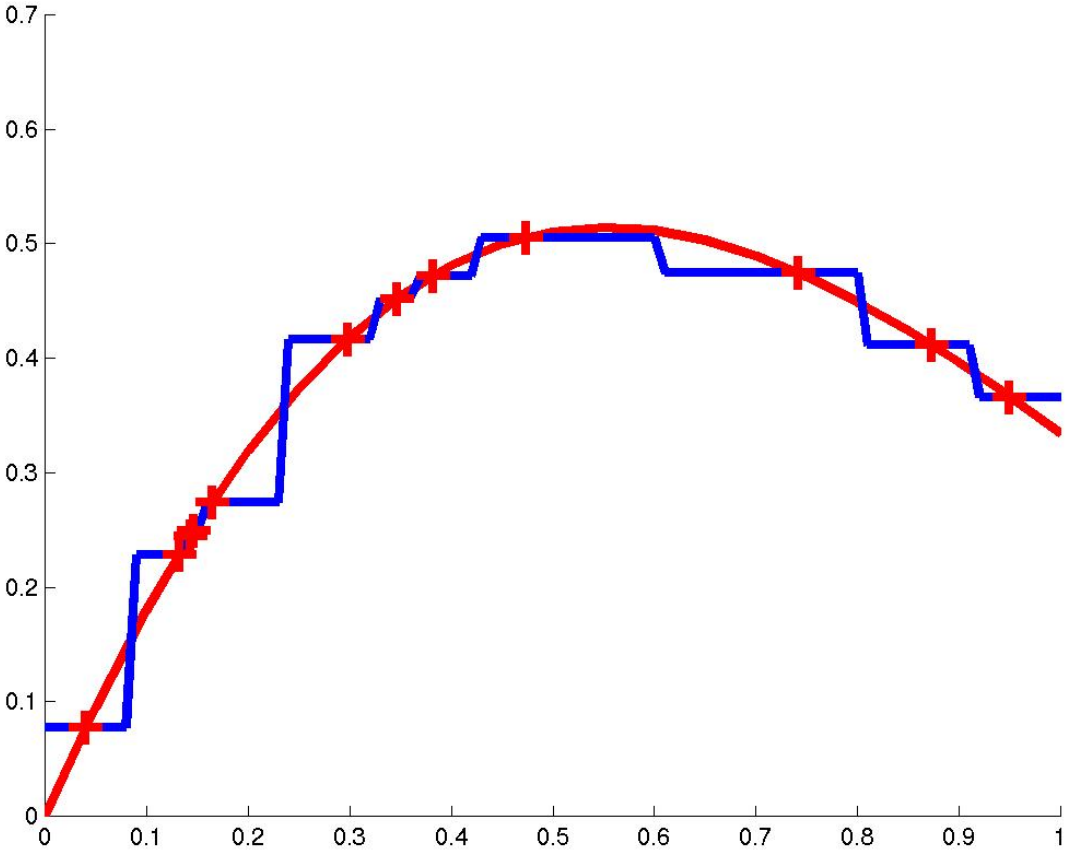
7 nearest neighbors

3x,4o



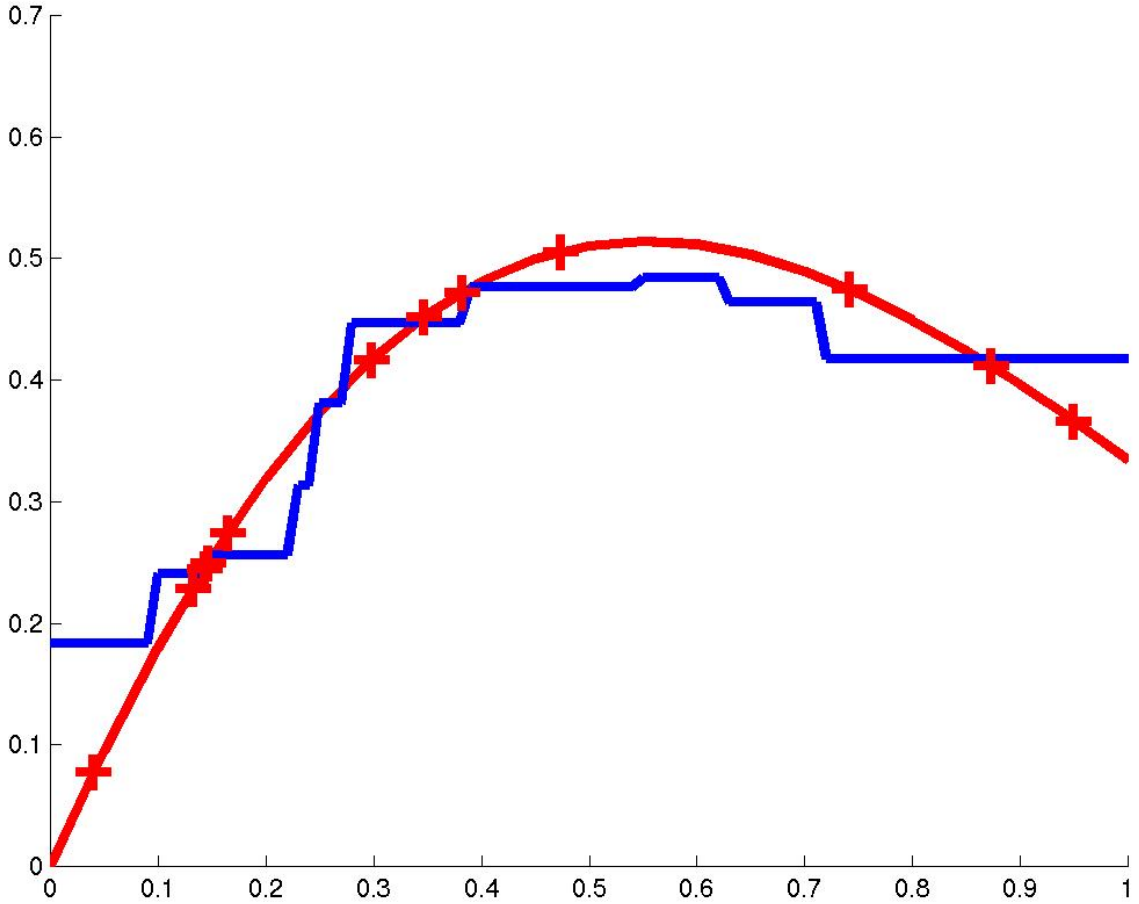
Nearest Neighbor (continuous)

1-nearest neighbor



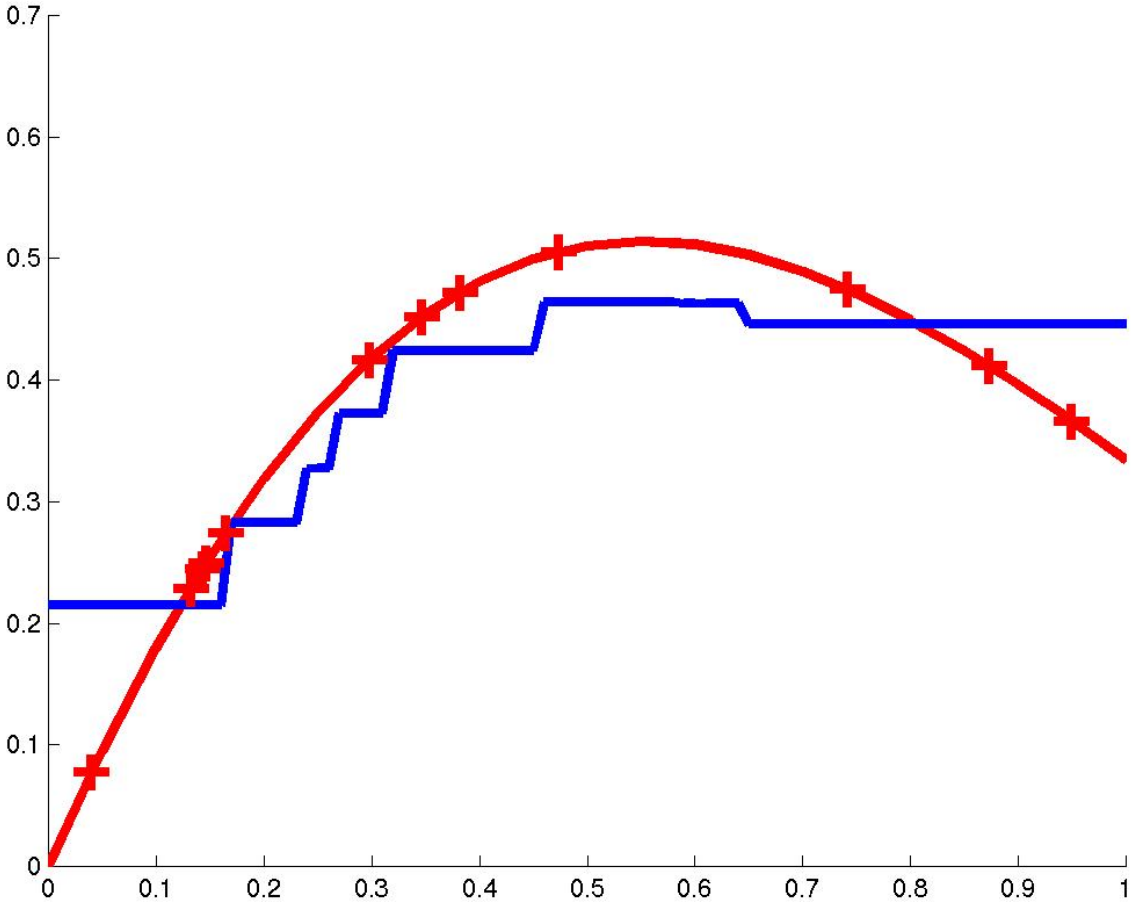
Nearest Neighbor (continuous)

3-nearest neighbor



Nearest Neighbor (continuous)

5-nearest neighbor



Distance Measures

- Euclidean distance
- Hamming distance
- City-Block distance
- Square distance
- Manhattan distance
- ...

Curse of Dimensionality

Imagine instances described by 20 attributes but only 2 are relevant to target function

Curse of dimensionality: nearest neighbor is easily misled when instance space is high-dimensional

One approach:

- Stretch j-th axis by weight z_j, where $z_1,\ldots,z_n$ chosen to minimize prediction error
- Use cross-validation to automatically choose weights $z_1,\ldots,z_n$
- Note setting zj to zero eliminates this dimension alltogether (feature subset selection)

Lazy and Eager Learning

- Lazy: wait for query before generalizing
  - k-nearest neighbors, weighted linear regression
- Eager: generalize before seeing query
  - Radial basis function networks, decision trees, back-propagation
- Eager learner must create global approximation
- Lazy learner can create local approximations
- If they use the same hypothesis space, lazy can represent more complex functions (H=linear functions)

Reading Assignment

- What are the strengths of the k-NN
- What are the limitations of the k-NN
- What is Case based reasoning? How does it work?